Formal Methods for Autonomic and Swarm-based Systems

Christopher Rouff
Amy Vanderbilt
SAIC
703-676-6184
rouffc@saic.com

Mike Hinchey
NASA GSFC
Code 581
michael.g.hinchey@nasa.gov

Walt Truszkowski, James Rash
NASA GSFC
Code 588
walter.f.truszkowski@nasa.gov
james.l.rash@nasa.gov

Abstract

Swarms of intelligent rovers and spacecraft are being considered for a number of future NASA missions. These missions will provide NASA scientist and explorers greater flexibility and the chance to gather more science than traditional single spacecraft missions. These swarms of spacecraft are intended to operate for large periods of time without contact with the Earth. To do this, they must be highly autonomous, have autonomic properties and utilize sophisticated artificial intelligence. The Autonomous Nano Technology Swarm (ANTS) mission is an example of one of the swarm type of missions NASA is considering. This mission will explore the asteroid belt using an insect colony analogy cataloguing the mass, density, morphology, and chemical composition of the asteroids, including any anomalous concentrations of specific minerals. Verifying such a system would be a huge task. This paper discusses ongoing work to develop a formal method for verifying swarm and autonomic systems.

Key Words: Swarms, autonomy, autonomic, asteroid, spacecraft, formal methods.

1. Introduction

Swarm technologies, whereby federated systems of spacecraft or rovers (of varying degrees of collective intelligence) mimic the societal behaviors of swarms, colonies, or flocks in nature (such as of bees, ants, or geese) appear to offer great potential, and are becoming a major focus for future NASA missions. These types of missions provide greater flexibility and the chance to gather more science than traditional single vehicle missions [6]. The emergent and autonomic properties of these missions make them powerful, but at the same time more difficult to design and verify. These missions are also more complex than previous types of missions, and NASA (or anyone else) has little experience in developing, verifying and validating them.

Bonabeau et al. [3] who has studied self-organization in social insects stated ``that complex collective behaviors may emerge from interactions among individuals that exhibit simple behavior'' and described emergent behavior as ``a set of dynamical mechanisms whereby structures appear at the global level of a system from interactions among its lower-level components.'' These emergent behaviors are the sum of simple individual behaviors, but when aggregated together form complex and often unexpected behaviors. Intelligent swarms [2] are where the individual members of the swarm have independent intelligence. This makes verification more difficult since swarm members are not homogeneous with limited functionality and communications.

For swarm exploration, individual autonomy is not crucial, but the mission cannot succeed unless each *team* has all the autonomic properties of being [11]. There are four such properties, which by their nature do not have clear boundaries:

- *self-configuring*, able to adapt to changes in the system;
- *self-optimizing*, able to improve performance;
- *self-healing*, able to recover from errors/damage; and
- *self-protecting*,-able to anticipate and cure intrusions.

The vision of Autonomic Computing as given in [11] views an autonomic system as being robust across these complementary dimensions.

Swarm-based systems will naturally bear all the hallmarks of a complex system – perhaps millions of lines of code, complex hardware-software interactions, real-time behavior, the necessity for continual updates, and a domain that is not fully understood. More importantly, such a system can never be properly or exhaustively tested. With the large number of parallel and distributed swarm members, the state space is extremely large and is impossible to test every pass through the state space.

Our conclusion is that having a formal model of these swarm missions will significantly help us verify that these systems can, and will, work properly. Formal methods are proven techniques for verifying complex systems, but due to the nature of swarm technologies, current methods must be modified or new methods must be created to

properly take into account the learning, intelligence and emergent behavior of such systems.

2. ANTS Mission Overview

The Autonomous Nano-Technology Swarm (ANTS) mission [6] will have swarms of autonomous pico-class (approximately 1kg) spacecraft that will search the asteroid belt for asteroids that have specific characteristics (Figure 1). There will be approximately 1,000 spacecraft involved in the mission. Present thinking has the swarm broken into three distinct classes: workers, which will carry high-end miniature instruments; others will be leaders that will be goal oriented and direct the workers and still others will be messengers that will route communications between leaders, workers and Earth. To examine an asteroid, the spacecraft will have to cooperate since they each only have a single instrument on board. To do this they will use an insect analogy of hierarchical social behavior were some spacecraft are directing others. Sub-swarms will exist that will act as teams that explore a particular asteroid based on the asteroids properties and share resources (instruments) between them.

To implement this mission a high degree of autonomy is being planned, approaching total autonomy, and will require autonomic properties. A heuristic approach is being considered that provides for a social structure to the spacecraft based on the above hierarchy. Artificial intelligence technologies such as genetic algorithms, neural nets, fuzzy logic and on-board planners are being investigated to assist the mission to maintain a high level of autonomy. Crucial to the mission will be the ability to modify its operations autonomously to reflect the changing nature of the mission and the distance and low bandwidth communications back to Earth.

3. Approaches and Assurance

As mission software becomes increasingly more complex, it also becomes more difficult to test and find errors. This is especially true of highly parallel processes and distributed computing, such as swarms and autonomic systems. Race conditions in these systems can rarely be found by inputting sample data and checking if the results are correct. These types of errors are time-based and only occur when processes send or receive data at particular times or in a particular sequence or after learning occurs. To find these errors, the software processes involved have to be executed in all possible combinations of states (state space) that the processes could collectively be in. Because the state space is exponential to the number of states, it becomes untestable with a relatively small number of processes. Traditionally, to get around the state explosion problem, testers have artificially reduced the number of states of

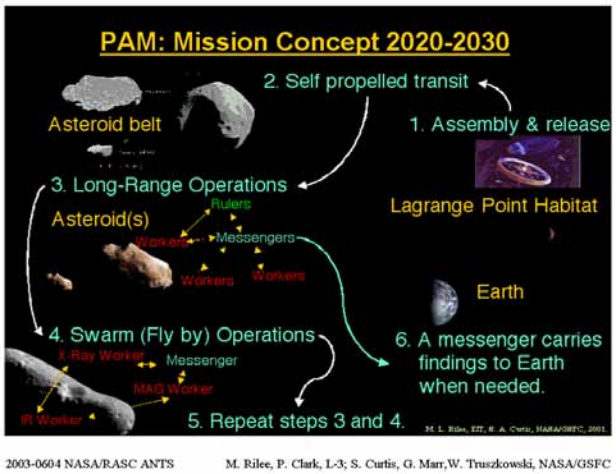

Figure 1: ANTS Mission Concept.

the system and approximated the underlying software using models.

Formal methods are proven approaches for assuring the correct operation of complex interacting systems [7, 12, 13]. They are particularly useful for specifying complex parallel and distributed systems where more than one person was involved in the development. Once written, a formal specification can be used to prove properties of a system correct, check for particular types of errors (e.g. race conditions), as well as used as input to a model checker. Verifying emergent behavior is one area that most formal methods have not addressed.

We surveyed formal methods techniques to determine if there existed formal methods that would be suitable for verifying swarm-based systems and their emergent behavior. It was found that there are a number of formal methods that support either the specification of concurrency or algorithms [14]. Though there were a few formal methods that have been used to specify swarm-based systems, only two formal approaches had been found that were used to analyze the emergent behavior of swarms. Weighted Synchronous Calculus of Communicating Systems (WSCCS), a process algebra, was used by Tofts to model social insects [17], and to analyze the non-linear aspects of social insects [16]. X-Machines have been used to model cell biology [9] and modifications have potential for specifying swarms. Simulation approaches are being investigated to determine emergent behavior. These approaches do not predict emergent behavior from the model but model the emergent behavior after the fact.

4. Specifications and Evaluation

In the initial evaluation of specification techniques for swarm-based systems [15], specifications of the NASA ANTS mission was done using Communicating Sequential Processes (CSP) [8], WSCCS, Unity Logic [4]

and X-Machines. Here we provide partial specifications of ANTS using the four methods, an evaluation of these methods and their potential for analyzing emergent behavior. In each case, only enough of the ANTS mission was specified to gather enough information to evaluate the method for specifying swarm-based systems. The following are the above specifications.

4.1. CSP

Each of the spacecraft has goals to fulfill their mission. The emergent behavior of all these goals should equal the goals of the mission. The following is the top-level specification of the ANTS mission:

$$ANTS_{goals} = Leader_{i,l\_goals} \,||\, Messenger_{j,m\_goals} \,||\, Worker_{k,w\_goals} \bullet 1 \le i \le m,\ 1 \le j \le n,\ 1 \le k \le p$$

where *m* is the number of leader spacecraft, *n* the number of messenger spacecraft and *p* the number of worker spacecraft. The ANTS mission starts, or is initialized, with a set of goals given to it by the principal investigator and part of these goals are given to the leader (some of these goals may not be given to the leader because the goals are ground based or not applicable to the leader). The leader spacecraft specification consists of two processes:

$$Leader_i = LEADER\_COM_{i,\{\}} \,||\, LEADER\_INTELLIGENCE_{i,goals,model}$$

the communications process and the intelligence process. The communication process, *LEADER_COM*, specifies the behavior of the spacecraft as it relates to communicating with the other spacecraft and Earth, and specifies a protocol between the spacecraft. The second process, *LEADER_INTELLIGENCE*, is the specification of the intelligence of the leader. This is where the deliberative and reactive parts of the intelligence are implemented and the maintenance of the goals for the leader is done. In addition to the goals, the *LEADER_INTELLIGENCE* process also maintains the models of the spacecraft and its environment and specifies how it is modified during operations. The following is an example portion of a top level specification of the leader communication:

$$LEADER\_COM_{i,conv} = leader.in?msg \rightarrow$$
case $LEADER\_MESSAGE_{i,conv,msg}$ *if* sender *(msg)* = $LEADER\_MESSENGER\_MESSAGE_{i,conv,msg}$ *if*
sender *(msg)* = $MESSENGER, WORKER\_MESSAGE_{i,conv,msg}$
if sender *(msg)* = $WORKER, EARTH\_MESSAGE_{i,conv,msg}$
if sender *(msg)* = $EARTH, ERROR\_MESSAGE_{i,conv,msg}$
otherwise

4.2. WSCCS

To model the ANTS Leader spacecraft, WSCCS (Weighted Synchronous Calculus of Communicating Systems), a process algebra, takes into account:

- The possible states (agents) of the Leader
- Actions each agent-state may perform that would qualify them to be in those states
- The relative frequency and priority of each action

Agent states and view of priority (p) and frequency (f) on the actions of the Leader as seen in Table 1. Based on this, the states of the Leader can now be defined by definition statements such as the following:

$$\begin{aligned} &Communicating \equiv \\ &50\omega^2 : ReasoningDeliberatve.Reasoning + \\ &50\omega^2 : ReasoningReactive.Reasoning \\ &+17\omega^2 : ProcessingSortingAndStorage.Proces sing \\ &+17\omega^2 : ProcessingGeneration.Proces sing \\ &+17\omega^2 : ProcessingPrediction.Proces sing \\ &+16\omega^2 : ProcessingDiagnosis.Proces sing \\ &+16\omega^2 : ProcessingRecovery.Proces sing \\ &+17\omega^2 : ProcessingRemediation.Proces sing \end{aligned}$$

This statement is saying that Leader, when in a Communicating state, has the option (is allowed) to perform any action from the set

$$\begin{aligned} &\{ReasoningDeliberatve, ReasoningReactive, \\ &ProcessingSortingAndStorage, \\ &ProcessingGeneration, ProcessingPrediction, \\ &ProcessingDiagnosis, ProcessingRecovery, \\ &ProcessingRemediation\} \end{aligned}$$

Table 1: Leader States and Actions

State	Action	f	p
	Identity		
Commun-icating	SendMessageWorker	50	2
	SendMessageLeader	50	2
	SendMessageError	1	1
	ReceiveMessageWorker	50	2
	ReceiveMessageLeader	50	2
	ReceiveMessageError	1	1
Reasoning	ReasoningDeliberatve	50	2
	ReasoningReactive	50	2
Processing	ProcessingSortingAndStorage	17	2
	ProcessingGeneration	17	2
	ProcessingPrediction	17	2
	ProcessingDiagnosis	16	2
	ProcessingRecovery	16	2
	ProcessingRemediation	17	2

and that the Communicating Leader will perform ReasoningDeliberatve with a probability of 25% and will give that action the same priority as the others. The second term in the statements tells us that the Communicating Leader will perform ReasoningReactive with the same 25% probability and priority of 2. The symbol + in this notation denotes a choice between the allowed actions, and the choice will be made based on the frequencies and priorities of each allowable action.

The single Leader by itself shows the following example emergent behavior. The Communicating Leader will choose to transition to a Processing state with a probability of 50% by choosing to process by one of the sic available processing types. It will choose from the six types with equal probability.

To study the emergent behavior of a swarm of Leaders we begin by considering a swarm of only 2 Leader spacecraft; called L1 and L2. Both leaders tick forward by performing one action per time step. Thus the two Leaders perform a composition of two actions, denoted $m1\omega^{k1} * m2\omega^{k2}$, on each time step. When this happens, the pair of leaders behaves according to the rules for composition:

$$n\omega^{k+l} * m\omega^{k} = (nm)\omega^{k+(k+l)} = m\omega^{k} * n\omega^{k+l}$$
$$n\omega^{k} * m\omega^{k} = (nm)\omega^{k+k} = m\omega^{k} * n\omega^{k}$$

This gives the Leader pair their own set of relative frequencies and priorities. Since there are two Leaders and each has three states and 14 possible actions, the pair of leaders has 9 possible state pairs and 196 possible action compositions. The 2-Leader swarm will have a much higher probability of having both leaders communicating or reasoning, rather than processing. Processing will be done by the swarm, but with much less frequency than communicating or reasoning. These features can be extrapolated to a swarm of n leaders as follows.

Given a swarm of n Leader Spacecraft, the n-leader swarm will tick forward in time by performing simultaneous actions – one action per leader per time step. Thus the n-leader swarm will perform (on each time step) a composition of n actions, denoted with weight $m_1\omega^{k_1} * m_2\omega^{k_2} * \ldots * m_n\omega^{k_n}$. When this happens, the n-leader swarm still must behave according to the rules for composition seen before.

This gives the n-leader swarm its own set of relative frequencies and priorities. Since there are n Leaders and each has three states and 14 possible actions, the swarm of n leaders has $\mathbf{3}^{n}$ possible state sets and $\mathbf{14}^{n}$ possible action compositions. There are only two possible priority values and four possible relative frequency values available and thus we can narrow down that each priority k_i must be either 1 or 2 and each relative frequency m_i must be either 1 (if the priority is 1) or one of 16, 17 or 50 (if the priority is 2). Thus the remaining options for leaders in the swarm will include communicating, reasoning, and processing (either by prediction or recovery, or otherwise). Let N_{comm} be the number of leaders in the swarm who choose to communicate (not in error) on a given time step. Let N_{reason} be the number of leaders in the swarm who choose to reason on that time step. Let $N_{process16}$ be the number of leaders in the swarm who choose to process (by prediction or recovery) on that time step. Lastly, let $N_{process17}$ be the number of leaders in the swarm who choose to process (by other means) on that time step. Then, each action by each leader will have priority 2 and relative frequency 16, 17 or 50. Thus, the composition of their actions will have weight:

$$m_1\omega^{k_1} * m_2\omega^{k_2} * \ldots * m_n\omega^{k_n} =$$
$$(50^{N_{comm}+N_{reason}})(16^{N_{process\,16}})(17^{N_{process\,17}})\omega^{2n}$$

From this weighting, we can see that drastically higher frequencies exist when larger numbers of the leaders in the swarm choose to communicate or reason. Much lower frequencies exist when larger numbers of leaders choose to process. Thus the swarm will be communicating and reasoning much more often than processing, although processing will take place.

4.3. Unity Logic

To model the ANTS Leader spacecraft with Unity Logic, we consider states of the Leader. In Unity Logic,

we will consider the states of the Leader, and the actions taken to make the Leader be in those states, but the notation will appear much closer to classical logic. Predicates will be defined to represent the actions that would put the Leader into its various states. Those predicates then become statements which, if true, would mean that the Leader had performed an action that put itself into the corresponding state. The Leader program would then be specified using ***assertions*** such as the following for Communication:

[Communicating]ReasoningDeliberatve(Leader)[Reasoning]
[Communicating]ProcessingGeneration(Leader)[Processing]

Unity Logic then provides a logical syntax equivalent to Propositional Logic for reasoning about these predicates and the states they imply as well as for defining specific mathematical, statistical and other simple calculations to be performed.

4.4. X-Machines

To model the ANTS Leader spacecraft as an X-Machine we must be able to see the Leader as a tuple:
$L = \{Input, Memory, Output, Q, \Phi, F, start, m_0\}$
where the components of the tuple are defined as

$$Input = \begin{Bmatrix} wor\,\mathrm{ker}, messenger, leader, error, \\ Deliberative, \mathrm{Re}\,active, \\ SortAndStore, \\ Generate, \mathrm{Pr}\,edict, Diagnose, \\ \mathrm{Re}\,\mathrm{cov}\,er, \mathrm{Re}\,mediate \end{Bmatrix}$$

$Memory$ will be written as a tuple $m = (Goals, Model)$ where Goals describes the goals of the mission and Model describes the model of the universe maintained by the Leader. The initial memory will be denoted by $(Goals_0, Model_0)$. When the goals and/or model changes, the new tuple will be denoted as $m' = (Goals', Model')$.

$$Output = \begin{Bmatrix} SentMessag\ eWor\,\mathrm{ker}, \\ SentMessag\ eMessenger, SentMessag\ eLeader, \\ SentMessag\ eError, \mathrm{Re}\,ceivedMess\ ageWor\,\mathrm{ker}, \\ \mathrm{Re}\,ceivedMess\ ageMesseng\ er, \\ \mathrm{Re}\,ceivedMess\ ageLeader, \\ \mathrm{Re}\,ceivedMess\ ageError, \\ \mathrm{Re}\,asonedDeli\ bartively, \mathrm{Re}\,asoned\ \mathrm{Re}\,actively, \\ \mathrm{Pr}\,ocessedSor\ tingAndSto\ ring, \\ \mathrm{Pr}\,ocessedGen\ eration, \mathrm{Pr}\,ocessed\ \mathrm{Pr}\,ediction, \\ \mathrm{Pr}\,ocessedDia\ gnosis, \mathrm{Pr}\,ocessed\ \mathrm{Re}\,\mathrm{cov}\,ery, \\ \mathrm{Pr}\,ocessed\ \mathrm{Re}\,mediation \end{Bmatrix}$$

$Q = \begin{Bmatrix} Start, Comm \\ \mathrm{Re}\,asoning, \end{Bmatrix}$ is a set of states.

$\Phi = \begin{Bmatrix} SendMessa \\ \mathrm{Re}\,ason, \mathrm{Pr} \end{Bmatrix}$ is a set of (partial) transition functions where each transition function maps $Memory \times Input \to Output \times Memory$ as in the following:

$$\Phi(m, Wor\,\mathrm{ker}) = (m', SentMessageWor\,\mathrm{ker})$$
$$\Phi(m, Generate) = (m', \mathrm{Pr}\,ocessedGeneration)$$

Then $F : Q \times \Phi \to Q$ is defined according to definitions such as in Table 2.

Table 2. Leader States and Transitions

Q	Φ	$Q' = F(Q, \Phi)$
Start	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Commun.	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Reasoning	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Processing	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing

5. Evaluation of Methods

CSP is very good at specifying the protocols between and within the spacecraft and analyzing the result for race conditions, which is very important in highly parallel systems, such as swarms. From a CSP specification, reasoning about the specification can be done to determine race conditions as well as converted into a model checking language for running on a model checker.

WSCCS also provides a process algebra that takes into account the priorities and probabilities of actions performed by the spacecraft. It also provides syntax and a set of rules for predicting and specifying choices and behaviors, as well as a congruence and syntax for determining if two automata are equivalent. All of this in hand, WSCCS can be used to specify the ANTS spacecraft and to reason about and even predict the behavior of one or more spacecraft. This robustness affords WSCCS the greatest potential for specifying emergent behavior in the ANTS swarm. What it lacks towards that end is an ability to track the goals and model of the ANTS mission in a memory. This may be achieved by blending the WSCCS methods with the memory aspects of X-Machines.

Unity Logic provides a logical syntax equivalent to simple Propositional Logic for reasoning about predicates and the states they imply as well as for defining specific mathematical, statistical and other simple calculations to be performed. However, it does not appear to be rich enough to allow ease of specification and validation of more abstract concepts such as mission goals. However, it

may be good for specifying and validating the Reasoning programming (as opposed to Reasoning process) portion of the ANTS Leader spacecraft, when the need arises.

X-Machines allow for a memory to be kept and it allows for transitions between states to be seen as functions involving inputs and outputs. This allows us to track the actions of the ANTS spacecraft as well as write to memory any aspect of the goals and model. This ability makes X-Machines highly effective for tracking and affecting changes in the goals and model. However, X-Machines do not provide any robust means for reasoning about or predicting behaviors of one or more spacecraft, beyond standard propositional logic.

6. Conclusion

An effective formal method must be able to predict the emergent behavior of 1000 agents as a swarm as well as the behavior of the individual agent. Crucial to the mission will be autonomic properties and the ability to modify operations autonomously to reflect the changing nature of the mission. For this, the formal specification will need to be able to track the goals of the mission as they change and to modify the model of the universe as new data comes in. The formal specification will also need to allow for specification of the decision making process to aid in the decision of which instruments will be needed, at what location, with what goals, etc.

Once written, the formal specification must be able to be used to prove properties of the system correct, check for particular types of errors (e.g. race conditions), as well as be used as input to a model checker. The formal method must also be able to track the models of the leaders and it must allow for decisions to be made as to when the data collected has met the goals.

To accomplish the above, a blending of the above methods seems to be the best approach for specifying swarm-based systems (Figure 2). Blending the memory and transition function aspects of X-Machines with the priority and probability aspects of WSCCS and other methods may produce a specification method that will allow all the necessary aspects for specifying emergent behavior in the ANTS mission and other swarm-based systems. The merging of these formal methods is currently being performed.

7. Acknowledgements

This work was supported by the NASA Office of Safety and Mission Assurance (OSMA) Software Assurance Research Program (SARP) and managed by the NASA Independent Verification and Validation (IV&V) Facility.

8. References

[1] ANTS Mission Web Site. NASA Goddard Space Flight Center. http://ants.gsfc.nasa.gov/

[2] Beni, G. and Want, J. Swarm Intelligence. In Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan, pp 425-428, Tokyo, Japan, 1989, RSJ Press.

[3] Bonabeau, E., G. Theraulaz, et al. Self-organization in Social Insects, Trends in Ecology and Evolution, 1997, vol. 12, pp. 188-193.

[4] Chandy, K. M. and Misra, J. Parallel Program Design: A Foundation. Addison-Wesley. 1988.

[5] Curtis, S. A., J. Mica, J. Nuth, G. Marr, M. Rilee, and M. Bhat. ANTS (Autonomous Nano-Technology Swarm): An Artificial Intelligence Approach to Asteroid Belt Resource Exploration. Int'l Astronautical Federation, Oct. 2000.

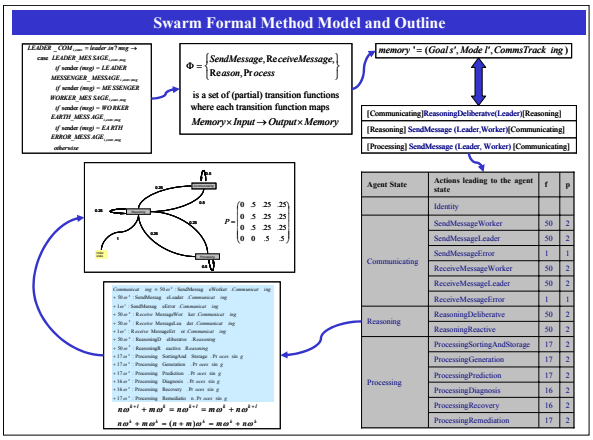

Figure 2: Combined formal method.

[6] Clark, P. E., Curtis, S. A. and Rilee, M. L. ANTS: Applying a New Paradigm to Lunar and Planetary Exploration. Solar System Remote Sensing Symposium, Pittsburg, 2002.

[7] Hinchey, M. and Bowen, J. Industrial-Strength Formal Methods in Practice. Springer. 1999.

[8] Hoare, C.A.R. Communicating Sequential Processes. Communications of the ACM, 21(8):666-677, Aug., 1978.

[9] Holcombe, M. Mathematical models of cell biochemistry. Technical Report CS-86-4. 1986. Dept of Computer Science, Sheffield University, United Kingdom.

[10] Holzmann, H. J, Design and Validation of Computer Protocols, Prentice Hall, Englewood Cliffs, NJ, 1991.

[11] Joseph, J. and Fellenstein, C. Grid Computing. IBM Press 2004.

[12] Nayak, P. Pandurang, et. al. 1999. Validating the DS1 Remote Agent Experiment. In Proceedings of the 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS-99).

[13] Rouff, C., Rash, J., Hinchey, M. Experience Using Formal Methods for Specifying a Multi-Agent System. Sixth IEEE Int'l Conference on Engineering of Complex Computer Systems (ICECCS 2000) September 11-15, 2000.

[14] Rouff, C., Vanderbilt, A., Truszkowski, W., Rash, J. and Hinchey, M. Verification of NASA Emergent Systems.

Ninth Int'l Conf. on Engin. of Complex Computer Systems (ICECCS 2004), Florence, Italy, April 14-16, 2004.

[15] Rouff, C., Vanderbilt, A., Hinchey, M. Truszkowski, W., and Rash, J. Properties of a Formal Method for Prediction of Emergent Behaviors in Swarm-based Systems. 2nd IEEE International Conference on Software Engineering and Formal Methods. Beijing, China, 26-30 September, 2004.

[16] Sumpter, D.J.T., Blanchard, G.B. and Broomhead, D.S. Ants and Agents: A Process Algebra Approach to Modelling Ant Colony Behaviour. Bulletin of Mathematical Bilogy. 2001, 63, 951-980.

[17] Tofts, C. Describing social insect behaviour using process algebra. Transactions on Social Computing Simulation. 1991. 227-283.